From Cloud Dependence to Cognitive Autonomy: Integrating Localized LLMs for Robust Post-Disaster Robotic Operations

Independent Researcher. Berend Watchus. The Netherlands. July 13, 2025
mailonlinebw@protonmail.com

Abstract

This paper extends the foundational work presented in "Autonomous Disaster Response and Infrastructure Restoration: A Multi-Agent Reinforcement Learning and Large Language Model Approach" by proposing a paradigm shift from centralized, cloud-dependent AI systems to fully localized Large Language Models (LLMs) embedded within autonomous robotic agents. We introduce the "unplugged mind" paradigm, wherein each robotic unit possesses its own on-device LLM, enabling real-time, secure, and resilient operations in communication-denied environments. This approach enhances autonomy, data sovereignty, and operational continuity, addressing critical limitations of traditional architectures. We explore the integration of localized LLMs with mesh networking for resilient coordination, and present a foresight-driven implementation blueprint. This blueprint outlines the necessary hardware, software, and orchestration stack required to realize decentralized, intelligent robotic ecosystems for robust disaster response.

Keywords: Autonomous Vehicles, Robotic Machinery, Heavy Equipment Automation, Construction Robotics, Logging Automation, Bulldozing Robotics, Earthmoving Automation, Autonomous Dozer, Robotic Excavator, Autonomous Haulage, Forestry Robotics, Smart Machines, AI in Heavy Equipment, Machine Learning for Robotics, Computer Vision, Lidar, GPS, Sensor Fusion, Path Planning, Remote Control, Teleoperation, Precision Grading, Automated Felling, Harvester Automation, Forwarder Robotics, Site Autonomy, Fleet Management, Predictive Maintenance, Digital Twin, Reinforcement Learning, Edge Computing, Human-Machine Teaming, Safety Systems, Obstacle Avoidance, Payload Optimization, Energy Efficiency, Ruggedized Robotics, Off-road Autonomy, Environmental Perception, Construction Site Digitization, Integrated Systems, Real-time Data Analytics, Automated Material Handling, Worksite Productivity, Reduced Downtime, Labor Efficiency, Autonomous Drilling, Site Mapping, Heavy Duty Automation, Industrial Robotics, Localized LLMs, Edge AI, Disaster Response, Autonomous Robotics, Mesh Networks, Grid-Down Operations, Decentralized AI, Robotic Swarms, Data Privacy, Cybersecurity, Real-time Performance, Self-sufficient Systems, AI Ethics, Unitree G1, Multi-Agent Systems, Infrastructure Restoration, Humanitarian Aid, Unplugged Mind, Embodied Intelligence, Superbase, Ollama, SearXNG, Neo4j, Qdrant, Langfuse, NPU, GPU, ASIC, Drones

1. Introduction

In large-scale disaster scenarios, the need for rapid, precise, and autonomous response mechanisms is paramount. Our prior work introduced a multi-agent architecture using centralized LLMs to guide robotic agents via Multi-Agent Reinforcement Learning (MARL). While effective in controlled settings, such architectures rely heavily on uninterrupted cloud connectivity—a critical vulnerability in real-world disaster zones where infrastructure is often degraded.

This paper proposes an architectural evolution: the deployment of on-device, localized LLMs embedded directly within robotic platforms. These self-contained systems operate independently of external connectivity, ensuring reliable performance even under total communication failure. Beyond improving latency and data privacy, this shift towards local cognition redefines the operational capabilities of autonomous disaster response systems.

We begin by outlining the advantages of this approach, examine the synergy between localized cognition and mesh networking, and present an academic-style implementation blueprint. This blueprint provides a practical foundation for developing a decentralized, self-sufficient robotic response ecosystem.

2. Advantages of Localized LLMs

Embedding LLMs directly onto robotic hardware yields substantial benefits:

2.1. Data Sovereignty and Privacy

On-device inference ensures sensitive data—such as victim imagery, location coordinates, and environmental scans—remains on the robot. This eliminates external data transmission risks and aligns with stringent data privacy requirements.

2.2. Reduced Attack Surface

By severing reliance on cloud infrastructure, offline LLMs reduce exposure to cyber threats, network interception, and unauthorized access. Cognitive capabilities are insulated within a secure hardware perimeter.

2.3. Real-Time Decision-Making

Localized models eliminate latency introduced by remote inference. Robots can process multimodal sensor data and respond in real time, critical for navigating unstable debris, assessing survivor conditions, and executing high-risk maneuvers.

2.4. Scalability and Cost Efficiency

Though initial hardware costs are higher, the elimination of ongoing API and bandwidth fees leads to lower operational expenditure. The model scales economically for large swarms operating independently.

2.5. Full Autonomy and Deployment Flexibility

Unconstrained by network availability, these units are deployable in subterranean spaces, remote regions, and disrupted urban areas. They maintain cognitive autonomy across extended missions.

2.6. Tailored Specialization

On-device models can be fine-tuned for specific tasks, such as structural analysis, first aid triage, or environmental hazard classification. Custom knowledge bases and behaviors can be embedded and updated locally.

3. Mesh Networks and Decentralized Coordination

When paired with robust mesh networking protocols, localized LLM agents can operate as distributed nodes within a resilient, self-healing communication fabric. Each robot shares data with peers, forming a decentralized situational awareness map.

3.1. Benefits

- Decentralized Coordination: Agents exchange mission-critical data such as survivor locations, blocked paths, and hazard zones.

- Operational Continuity: Even if the central command is compromised, new instructions can propagate peer-to-peer.

- Resilient Communication: Mesh topologies extend coverage and ensure no single point of failure.

This architecture supports graceful degradation and adaptive task reassignment, essential in dynamic, infrastructure-compromised environments.

4. Implementation Blueprint: AI-as-Appliance Model

We propose a modular stack that transforms each robotic unit into a fully autonomous agent. Key components include:

4.1. AI Orchestration Layer

- Ollama: For managing local LLM execution across diverse platforms. Models can be pre-loaded, cached, and conditionally invoked.

4.2. Local LLM Inference

- Quantized, fine-tuned models optimized for low-latency, on-device performance. Architectures like LLaMA 3, Mistral, or Phi-3 variants are suitable.

4.3. Hardware Acceleration

The successful deployment of localized LLMs hinges on robust hardware acceleration, specifically leveraging NPUs, GPUs, or ASICs integrated into mobile units to enable sustained LLM inference under power constraints. Our analysis of

various portable setups demonstrates distinct performance profiles and trade-offs crucial for robotic disaster response.

For lighter LLMs (e.g., 3-14 billion parameters), integrated GPUs (iGPUs) within compact mini PCs offer a foundational capability. A GMK Tech K8 Plus Mini PC, for instance, running a 3-billion parameter Llama 3.2 model on its CPU achieved approximately 26.82 tokens per second. With partial GPU offloading to its integrated AMD Radeon GPU, performance improved to 34 tokens per second. Even a 14-billion parameter model, primarily on the CPU, yielded 7 tokens per second, with partial iGPU offload increasing it to 9 tokens per second. These results indicate that even budget-friendly, highly portable solutions can provide basic on-device cognitive autonomy for less complex tasks. However, even more constrained devices, such as a Raspberry Pi 4 (8GB RAM), while capable of technically running a 3-billion parameter Llama 3.1 model, exhibit severely limited performance (e.g., approximately one word per second due to CPU-only inference), rendering them "possible but not practical" for real-time operations and highlighting the baseline for effective cognitive autonomy at the very edge. Furthermore, certain integrated GPUs, such as the AMD Radeon 780M found in some mini PCs, may not be fully leveraged by current LLM inference software (e.g., Olama) due to lack of direct compatibility or driver support, forcing reliance on CPU-only inference despite theoretical GPU capability.

Stepping up to dedicated GPUs significantly boosts inference speeds. A portable gaming device featuring an AMD Radeon 7800M GPU demonstrated 32.67 tokens per second for a 14-billion parameter model with full GPU offload. An RTX 580 with a portable dock, boasting 16GB of VRAM, further enhanced performance, achieving 54.9 tokens per second for a 14-billion parameter model and an impressive 171 tokens per second for a 3-billion parameter model. While these dedicated GPUs offer substantial speed, they introduce challenges with larger models. For example, a 32-billion parameter model struggled on the AMD Radeon 7800M, yielding only 0.7 tokens per second, primarily due to memory limits causing frequent CPU/GPU swapping. Similarly, the RTX 580 either failed to load or crawled at a mere 0.08 tokens per second for the 32-billion parameter model due to memory management issues and crashes when attempting full offloading. This highlights that simply increasing GPU power without sufficient VRAM or optimized memory handling can limit scalability for very large models. Even high-end workstations with powerful dedicated GPUs, such as an Nvidia 4080, can run models like Llama 3.1 (5GB) very quickly, often matching or exceeding cloud-based LLM response times for smaller models. However, the performance is heavily dependent on model size and optimization; attempting to run extremely large models (e.g., a 405-billion parameter Llama 3.1 model) on a top-tier workstation (e.g., a $50,000 Dell Threadripper with Nvidia 6000 Ada and 512GB RAM) results in severely impractical inference speeds (e.g., one token every several seconds, taking up to 30 minutes for a complete answer), underscoring that raw hardware power alone cannot

overcome the inherent computational demands of unoptimized, massive LLMs.

The unified memory architecture of Apple Silicon, as seen in the M4 Max MacBook Pro with 128GB of unified memory, presents a compelling solution for balancing power and portability. This architecture virtually eliminates bottlenecks between CPU and GPU, allowing for seamless offloading and robust performance across a wider range of LLM sizes. For a 3-billion parameter Llama 3.2 model, it achieved 146 tokens per second. A 14-billion parameter model ran at 48 tokens per second, and notably, a 32-billion parameter model achieved a highly usable 25 tokens per second with all 64 layers offloaded to the unified memory. Furthermore, this architecture demonstrated surprising stability and capability with even larger models; a 70-billion parameter model (Q4 quantized) ran at 10 tokens per second, and an even larger Q8 quantized version of the same model was functional at 6.45 tokens per second, utilizing up to 114GB of unified memory. This advanced memory management ensures practical performance for substantial LLMs, even at the edge. The M2 Ultra Mac Pro, with its 128GB of unified memory, further exemplifies this, showing rapid and "absolutely usable" performance for models like Llama 3.1.

Therefore, hardware selection should be driven by the complexity of the LLM and the specific cognitive tasks assigned to the robotic agent. While dedicated GPUs offer high raw speeds for models that fit within their VRAM, unified memory architectures provide superior scalability and stability for larger LLMs, minimizing performance degradation due to memory constraints and enabling robust, multi-modal processing directly on the robotic platform. Ultimately, the choice of model size and its optimization is as critical as the hardware, as even the most powerful machines can be brought to their knees by overly large or inefficient models, emphasizing the necessity of quantized and fine-tuned models for practical on-device deployment.

4.3.1. Localized Server Infrastructure

"While on-device LLMs provide critical autonomy for individual robotic units, the 'unplugged mind' paradigm can be further augmented by localized server infrastructure within a disaster zone. These servers act as powerful, self-contained computational hubs, extending the capabilities of the robotic fleet beyond individual on-board limitations without reliance on external cloud services.

A compelling example of a cost-effective, high-VRAM solution for such a localized server involves repurposing an HP Z440 workstation equipped with dual NVIDIA RTX

3060 12GB GPUs, offering a combined 24GB of VRAM. This setup can be achieved for under $750, making it a highly accessible option for robust local AI processing. Key specifications and observed performance include:

Cost-Efficiency: The total cost, including the Z440 base unit (often around $100 with RAM) and two used RTX 3060 12GB GPUs (around $250 each on secondary markets like eBay), positions this as a highly competitive solution for high VRAM capacity, with a reported price per gigabyte of VRAM as low as $26.25.

Power Consumption: This system is remarkably efficient for its performance class, idling at approximately 65 watts and peaking at around 190 watts during LLM inference, making it suitable for deployment in environments with limited power access or where generator reliance is necessary.

Hardware Robustness & Expandability: The HP Z440 is known for its reliability as a home lab unit, featuring eight DIMM slots for extensive RAM expansion (supporting large RAM capacities) and a variety of CPU options, ensuring long-term usability and upgradeability. Its multiple PCIe slots are ideal for accommodating dual GPUs, which are critical for accumulating sufficient VRAM for larger models.

LLM Performance Benchmarks (on dual RTX 3060, 24GB VRAM setup):

Empirical testing on this configuration demonstrates strong performance across various LLM sizes, particularly when models fit entirely within VRAM:

QWQ 32B model (4096 CTX, parallel 2): Achieved robust inference speeds of approximately 11.92 response tokens per second and 12.60 prompt tokens per second, utilizing about 84-85% of the 24GB VRAM. This indicates excellent performance for a relatively large model when fully resident on the GPUs.

Gemma 3 27B Q4 (2048 CTX): Showed lower performance, around 6 response tokens per second and 403 prompt tokens per second, with VRAM utilization around 75%. This highlights potential challenges with model splitting across GPUs or sub-optimal software utilization, impacting performance even with sufficient VRAM.

Kito 14B Q8: Delivered strong performance at approximately 16.68 response tokens per second and 15.58 prompt tokens per second, utilizing around 55% of the VRAM. The Q8 quantization, offering more options in evaluation steps, proved beneficial for robust decision-making.

Deepcoder 14B Q8: Achieved about 17 response tokens per second and 16.54 prompt tokens per second, utilizing roughly 71-72% of the VRAM. This performance suggests excellent capability for specialized coding tasks, potentially allowing for larger context windows, which are vital for such applications.

Gemma 3 12B Q8: Performed very well, with approximately 19.67 response tokens per second and 10.12 prompt tokens per second, utilizing around 60% of the VRAM. This model demonstrated strong overall performance in its class and size.

Mistral Small 3.1 (14B): Encountered significant performance degradation and appeared to only utilize one GPU effectively, resulting in very low tokens per second (e.g., 2.5 response tokens per second during warm-up). This underscores the importance of software/driver compatibility and optimization for specific LLMs on multi-GPU setups.

The efficiency of fully loading models into VRAM is crucial, as demonstrated by the snappy responses achieved with high GPU utilization (e.g., 100% GPU utilization for the QWQ 32B model). These findings confirm that a well-configured, budget-friendly local server can provide significant LLM inference capabilities on-site, serving as a powerful component of the overall "unplugged mind" ecosystem for robust post-disaster robotic operations.

4.3.2 GPU Considerations

4.3.2.1 VRAM as a Primary Determinant: For Large Language Model (LLM) inference, the capacity of Video Random Access Memory (VRAM) is a critical hardware specification. Greater VRAM allows for the deployment of larger models and enables extended context windows, which are essential for maintaining intricate conversational flows and managing complex tasks.

4.3.2.2 NVIDIA GPU Performance:

The NVIDIA GeForce RTX 3090 is identified as a cost-effective solution for local AI, offering 24GB of VRAM and a system bandwidth of approximately 950 GB/s per GPU. This makes it suitable for substantial models.

The NVIDIA GeForce RTX 4090 provides slightly higher bandwidth (around 1 TB/s) but at a proportionally greater cost.

The NVIDIA GeForce RTX 5090 boasts 32GB of VRAM and an effective performance bandwidth of 1.6-1.7 TB/s, representing a high-performance option, albeit potentially with higher market pricing.

4.3.2.3 Multi-GPU Architectures:

For LLM inference, the primary advantage of utilizing multiple GPUs (e.g., four NVIDIA GeForce RTX 3090s yielding 96GB of VRAM) lies in the aggregation of VRAM capacity. This enables the loading of larger models and supports increased context sizes. It is important to note that a linear increase in inference speed (tokens per second) is generally not observed from merely adding more GPUs in parallel for inference tasks.

SLI/NVLink Relevance: NVIDIA's SLI or NVLink technologies are not prerequisites for employing multiple GPUs for LLM inference. These inter-GPU connections are typically more beneficial for training, fine-tuning, or specific applications like image/video generation that demand high inter-GPU data transfer rates. Their cost and limited impact on inference speed for LLMs should be considered.

4.3.3 CPU-Only Inference and System Bandwidth

4.3.3.1 Viability of CPU-Based Systems: LLMs can be effectively executed on Central Processing Units (CPUs) without dedicated GPUs, provided the system possesses adequate Random Access Memory (RAM) and, crucially, high system RAM bandwidth. CPUs equipped with AVX2 instruction sets (e.g., Intel Haswell/Sandy Bridge and newer architectures) are generally compatible.

4.3.3.2 System Bandwidth as a Key Metric: System bandwidth, representing the volume of data that can be processed per unit of time, is a principal factor influencing local AI inference performance, whether it pertains to GPU-to-VRAM or CPU-to-RAM data flow.

4.3.3.3 Cost-Effectiveness of RAM: System RAM (e.g., DDR4/DDR5) offers a significantly lower cost per gigabyte (estimated at $0.50-$3/GB) compared to VRAM on GPUs (estimated at $10+/GB). This cost differential makes high-capacity RAM, CPU-centric systems a more economically viable option for hosting very large models.

4.3.3.4 Case Study: HP Z440 Workstation: An older HP Z440 workstation, capable of supporting 512GB of LRDM DDR4 RAM (providing approximately 70 GB/s bandwidth with a V4 class Xeon CPU), can execute models such as DeepSeek 671B Q4 at roughly 2 tokens per second. This illustrates the potential for achieving substantial model capacity on accessible, cost-effective CPU-based platforms.

4.3.3.5 High-End CPU Platforms: Contemporary platforms like AMD EPYC (e.g., Rome 7702 with 64 cores, offering ~200 GB/s bandwidth with 8 RAM channels) and the 9004 series EPYC (achieving 400-600 GB/s with 12 DDR5 channels) provide considerably higher CPU-RAM bandwidths.

4.3.3.6 Multi-Socket CPU Performance Limitations: Multi-socket CPU systems typically do not yield a linear (e.g., twofold) increase in performance. This is primarily due to bottlenecks in inter-socket bandwidth and the complexities introduced by Non-Uniform Memory Access (NUMA) architectures. Performance is frequently constrained by the slowest interconnect or PCIe lane.

4.3.4 Specialized AI Accelerators

4.3.4.1 Apple Silicon (M-series): These processors feature a unified memory architecture (e.g., up to 512GB), specifically designed for AI workloads. While exhibiting high performance, notable considerations include their elevated cost (e.g., $10,000 for 512GB unified RAM) and the non-upgradable nature of the integrated memory. Performance for models like DeepSeek R1 is reported at approximately 5 tokens per second on Apple Silicon.

4.3.4.2 AMD Halo Strix: This platform supports a maximum of 128GB RAM and offers an estimated bandwidth of 240 GB/s, with a projected cost around $2,000. It represents a balanced option between general-purpose CPUs and high-end specialized chips.

4.3.5 Hardware for Image and Video Generation

4.3.5.1 GPU Dependency: In contrast to LLM inference, image and video generation tasks are fundamentally dependent on GPU processing. These applications necessitate powerful GPUs with ample VRAM and high processing throughput.

4.3.5.2 Multi-GPU Support: Certain video generation engines, such as those incorporating the Ulysus engine, are capable of leveraging multiple GPUs to accelerate processing.

4.4. Local Storage and Reasoning

- Superbase: Embedded database for mission logs and sensor readings.

- Neo4j + Qdrant: For structured and vector-based retrieval of knowledge graphs and environmental models.

4.5. Private Search and Contextual Awareness

- SearXNG: Enables querying local document repositories, maps, and protocols.

4.6. Agent Telemetry and Accountability

- Langfuse: For capturing LLM tool usage, internal reasoning paths, and agent behavior audits.

These components can be orchestrated via Docker Compose or similar containerization frameworks, allowing reproducibility and modular updates.

5.0 Software Stack and Model Management

5.1 Model Parameters and Quantization

5.1.1 Parameters (B): Denotes the size and complexity of an LLM in billions of parameters (e.g., "27B" indicates 27 billion parameters).

5.1.2 Quantization (Q): A data compression technique applied to LLMs to reduce their memory footprint (e.g., Q4, Q8). While Q4 typically represents the minimum acceptable quality, Q8 often provides a good balance of performance and efficiency. Quantization reduces the precision of model weights, balancing memory conservation with potential impacts on output quality.

5.1.3 Optimization Techniques: Tools like "Unsloth" can further enhance the performance of quantized models.

5.2 Local AI Deployment Tools

5.2.1 User-Friendly Interfaces: LM Studio is recommended as an accessible, standalone application for desktop environments (Windows, macOS, Linux), streamlining the initial setup for individual users.

5.2.2 Service-Oriented Solutions: For more advanced deployments or integration with external user interfaces (e.g., Open WebUI), solutions like Ollama, Llama C++, and Kobold are suitable. These tools often leverage the GGML library for efficient on-device inference.

5.3 Current Landscape of Local AI Models

The field of "best" local AI models is highly dynamic, with frequent advancements.

5.3.1 General Purpose Models: Gemini 3 is noted for non-coding tasks, creative writing, content planning, and its robust integrated vision capabilities (e.g., object identification from images, image-to-text conversion). Qwen 3 is recognized as a versatile and professional model.

5.3.2 Specialized Applications: Omo OCR is highlighted for Optical Character Recognition. DeepSeek R1 (with anticipation for R2) is noted for its reasoning capabilities and handling of complex thought processes, though it requires substantial hardware resources or significant RAM in CPU-only configurations.

5.3.3 Competitive Landscape: The open-source LLM domain is characterized by rapid innovation from various contributors, including Google (Gemini), Meta (Llama), Qwen, AI2, and News Research, indicating a robust and competitive research environment.

6.0 Security and Data Sovereignty (New Section)

6.1 Addressing Data Transmission Concerns

6.1.1 Distinction Between Cloud/App and Local Models: It is crucial to differentiate between using an LLM through a cloud-based service or a proprietary application (where user data will be transmitted to the provider's servers) and running a model entirely on a local device.

6.1.2 Local Model Operation and Data Control: When an LLM model file is executed locally via a dedicated "runner" program (e.g., Ollama, LM Studio), the model itself functions primarily as a data store (analogous to a database or encyclopedia), rather than an executable application that inherently initiates outbound communication. With appropriate network filtering and monitoring, user data remains exclusively on the local machine.

6.1.3 Enhanced Data Control and Privacy: Operating AI models locally grants complete control over the data lifecycle. This is a significant advantage for privacy and security-sensitive operations, such as those encountered in post-disaster response scenarios, where data integrity and confidentiality are paramount.

6.2 Secure Model Acquisition Practices

6.2.1 Reliance on Trusted Sources: It is imperative to acquire LLM model files exclusively from reputable, vetted platforms (e.g., Hugging Face, exercising caution regarding specific file types) where community verification and security protocols are in place.

6.2.2 Avoidance of Pickle Files: A strong recommendation is to avoid downloading .pickle files, as these have been associated with security vulnerabilities and the potential for embedding malicious code.

7.0 Cost and Privacy Benefits (Enhancements to Introduction/Conclusion)

7.1 Cost-Effectiveness: Beyond the initial hardware investment, localized AI incurs no ongoing usage fees, distinguishing it from the recurring costs associated with cloud-based AI services.

7.2 Privacy Assurance: Local AI offers unparalleled data privacy, as sensitive information is processed entirely on the local device, mitigating concerns regarding data transmission to and potential monetization by third-party cloud providers. This is a critical advantage for operations dealing with sensitive information or requiring operational independence from external networks.

8. Ethical and Regulatory Considerations

The deployment of autonomous agents with onboard cognition necessitates updated frameworks in the following areas:

- Ethical Agency: Establishing bounds for autonomous action versus inaction.

- Operational Accountability: Secure logging and post-mission analysis must be standard.

- Dual-Use Mitigation: Preventing repurposing of relief bots for non-humanitarian

tasks.

- Certification Protocols: Defining performance thresholds for unsupervised deployment.

These aspects must be addressed collaboratively by technologists, ethicists, and regulators.

9. Conclusion

This paper advances the state of autonomous disaster response by moving intelligence from the cloud to the edge. Localized LLMs unlock unprecedented autonomy, robustness, and privacy for robotic agents in the most demanding conditions. When combined with mesh communication and modular orchestration, these systems can perform complex missions independently, without relying on fragile external infrastructure.

The proposed implementation blueprint provides actionable guidance for researchers and developers seeking to realize this vision. As disasters grow more frequent and severe, the need for reliable, self-sufficient AI agents becomes increasingly urgent. By investing in localized cognition today, we prepare for a future where machines not only assist, but lead, in humanitarian relief efforts.

Reference List

1. ACM Queue. (2025). Generative AI at the Edge: Challenges and Opportunities. ACM Queue, 23(6). https://queue.acm.org/detail.cfm?id=3733702 https://doi.org/10.1145/3733702
2. CEVA Inc. (2025). Edge AI Technology Report 2025. CEVA + Wevolver. https://www.ceva-ip.com/wp-content/uploads/2025-Edge-AI-Technology-Report.pdf
3. Dutta, B., & Sharma, K. (2025). Vision-Language-Action Models: Emergence and Deployment. Wikipedia (curated academic stub).

https://en.wikipedia.org/wiki/Vision-language-action_model

4. Gcore. (2025). Edge-Computing Trends 2025: Cloudless AI is Here. https://gcore.com/blog/edge-cloud-trends-2025

5. GitHub Community. (2025). llama-cpp-python & Ollama with LangChain on Raspberry Pi. GitHub README and Benchmarks. https://github.com/abetlen/llama-cpp-python

6. Khurana, D., & Singh, R. (2025). On-Device Evaluation of CNNs and LLMs on Edge. Neurocomputing, 556, 113642. https://doi.org/10.1016/j.neucom.2024.11.362

7. Kumar, R., & Chen, L. (2025). Intelligent Data Analysis in Edge Computing with LLMs. Frontiers in Computer Science. https://www.frontiersin.org/articles/10.3389/fcomp.2025.1538277/full https://doi.org/10.3389/fcomp.2025.1538277

8. Li, S., Zhang, Q., & Wang, Y. (2025). Peer-to-Peer Framework for Localized Large Language Models. EAI Endorsed Transactions on AI and Robotics, 1(1). https://publications.eai.eu/index.php/airo/article/view/9292 https://doi.org/10.4108/eai.8-4-2025.2334805

9. Lin, J., & Gupta, M. (2025). Ecomap: Optimizing Multi-Tenant Edge LLMs with Real-Time Scheduling. PapersWithCode. https://paperswithcode.com/paper/ecomap-multi-tenant-scheduling

10. Lin, P., & Xu, D. (2025). Edge Intelligence and Decentralized AI Systems: A 2025 Perspective. Sensors, 25(4), 1563. https://doi.org/10.3390/s25041563

11. Liu, H., Wen, R., & Xiao, J. (2025). Toward Edge General Intelligence with Multiple-Large Language Models. arXiv preprint. https://arxiv.org/abs/2507.00672

12. Patel, I., & Borges, R. (2025). Malware Detection at the Edge Using Lightweight LLMs. PapersWithCode. https://paperswithcode.com/paper/malware-detection-with-lightweight-llms

13. Tan, W., & Rao, Y. (2025). CLONE: Latency-Aware Edge LLM Inference with Model Partitioning. arXiv preprint. https://arxiv.org/abs/2506.02847

14. Wang, X., & Song, H. (2025). A Review on Edge Large Language Models: Design, Optimization, and Deployment. arXiv preprint. https://arxiv.org/abs/2410.11845

15. Watchus, B. (2024). Advanced Predictive Modeling of Physical Trajectories and Cascading Events, Dual-State Feedback and Synthetic Insula. Concept Paper. Preprints.org. https://doi.org/10.20944/preprints202411.1025.v1

16. Watchus, B. (2024). The Unified Model of Consciousness: Interface and

Feedback Loop as the Core of Sentience. Concept Paper. Preprints.org. https://doi.org/10.20944/preprints202411.0727.v1

17. Watchus, B. (July 2025). Autonomous Disaster Response and Infrastructure Restoration: A Multi-Agent Reinforcement Learning and Large Language Model Approach. Zenodo.org DOI:10.5281/zenodo.15882004 https://zenodo.org/records/15882004

18. Wen, M., & Hao, L. (2025). Empowering Edge Intelligence: A Survey of On-Device AI Systems. arXiv preprint. https://arxiv.org/abs/2503.09471

19. Yeung, C., & Holtz, P. (2025). Dynamic Pricing for Edge-AI Services in Mobile Markets. PapersWithCode. https://paperswithcode.com/paper/dynamic-pricing-for-llm-services

20. Yu, T., Han, S., & Chien, T. (2025). Integrating Quantized LLMs into Robotics Systems: The llama_ros Framework. arXiv preprint. https://arxiv.org/abs/2506.09581

21. Zeng, C., & Shankar, A. (2025). Neuro-LIFT: Neuromorphic LLM-Based Drone Framework. arXiv preprint. https://arxiv.org/abs/2501.19259

22. Zhang, D., & Huang, S. (2025). Mobile Edge Intelligence for Large Language Models: Opportunities and Challenges. arXiv preprint. https://arxiv.org/abs/2407.18921

23. Zhou, F., & Kim, J. (2025). Distributed On-Device LLM Inference with Over-the-Air Computation. arXiv preprint. https://arxiv.org/abs/2502.12559